INSTITUTO FEDERAL
Ceará
Campus Tianguá

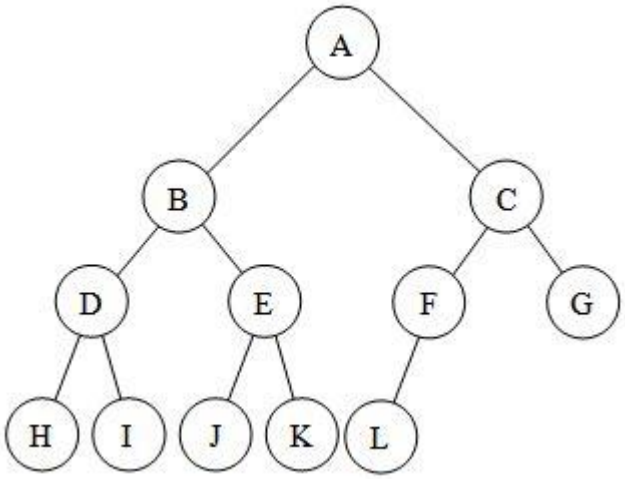
TRABALHO COMPUTACIONAL 02 – ÁRVORES E HEAPSORT	
CURSO	Bacharelado em Ciência da Computação
DISCIPLINA	Construção e Análise de Algoritmos
PROFESSOR	Adonias Caetano de Oliveira
EQUIPE	**NO MÁXIMO 04 PESSOAS**

INSTRUÇÕES:

I. O não cumprimento sobre a quantidade máxima de alunos por equipe resulta em -2 pontos na atividade;
II. Todos os códigos-fontes devem ser compactados em arquivo ".zip" ou ".rar". Se for um arquivo do Colab não precisa compactar.
III. A entrega do trabalho pode ser por anexo no classroom ou no e-mail ou mesmo compartilhado pelo Google Drive;
IV. Não compartilhe arquivos executáveis;
V. Respostas das questões teóricas precisam estar em um arquivo PDF;
VI. **Envie a imagem do gráfico gerado;**
VII. Proibido uso de bibliotecas ou APIs que implementam árvores ou métodos de ordenação;
VIII. Comente as linhas importantes dos códigos e seus métodos;
IX. Não será aceito nenhuma resposta das questões anotadas em folhas de caderno ou A4 ou impressa ou digitalizadas ou outro tipo não autorizado nestas instruções;
X. Compartilhar ou enviar ao e-mail: adonias.oliveira@ifce.edu.br
XI. Será atribuída a mesma nota para cada membro de uma mesma equipe. Caso o líder da equipe discorde disto deve executar as seguintes ações:
   - Enviar um e-mail com cópias para todos os membros da equipe;
   - No corpo do e-mail deve atribuir valores de 0% até 100% como nível de contribuição que cada membro deu no trabalho;
   - Descrever a metodologia de divisão das tarefas
   - Relatar os problemas que teve com outro membro que prejudicou o trabalho;
XII. Após a correção desta atividade, em caso de discordância da nota E negativa do professor em atender suas solicitações, é preciso seguir as orientações do artigo 96 do ROD do IFCE.

QUESTÕES TEÓRICAS

1) Dada a árvore abaixo, indique:



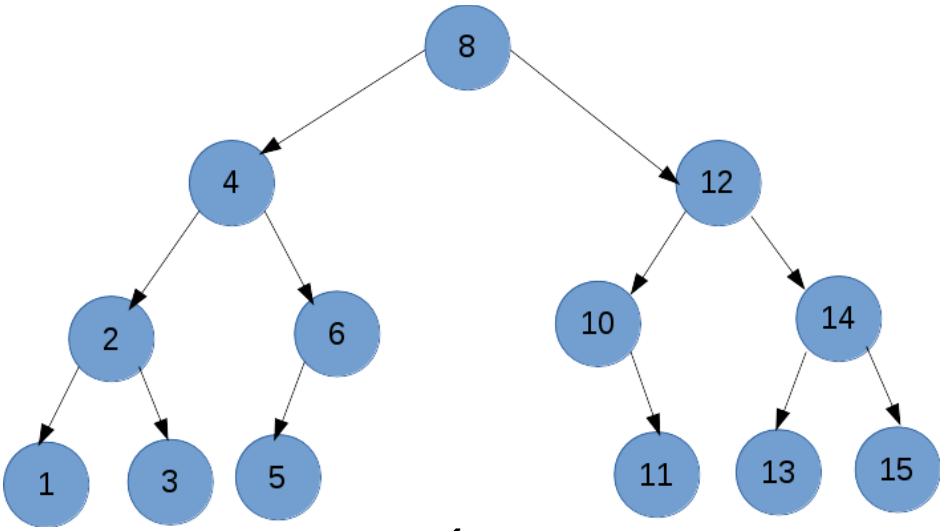
a) nós-folha;
b) grau de cada nó;
c) grau da árvore;
d) altura de cada nó;
e) altura da árvore;
f) a profundidade de cada nó;
g) a profundidade da árvore;
h) os descendentes do nó H.
i) Percurso pré-ordem.
j) Percurso em ordem.
k) Percurso pós-ordem.

2) Represente a árvore da questão 01 no formato:
a) Paragrafação
b) Parênteses aninhados
c) Diagramas de Venn
d) Diagrama de barras.

QUESTÕES PRÁTICAS

3) Implemente a estrutura de dados Árvore Binária em Python ou Java ou C++ usando conceitos de orientação a objetos. A sua classe deve ter os seguintes métodos:

   a) **getNosFolha**: retorna uma lista de nós folhas;
   b) **getGrau(int nó)**: retorna o grau de um nó;
   c) **altura**: retorna altura da árvore;
   d) **profundidade**: retorna o valor de profundidade da árvore;



Exemplo de Árvore Binária

4) Implemente em Python todos os dez algoritmos de ordenação ensinados em sala de aula, realizando experimentos que avaliem o tempo de execução para ordenar de acordo com as seguintes regras:

   I. Serão nove vetores com os seguintes tamanhos para cada um: 1000, 3000, 6000, 9000, 12000, 15000, 18000, 21000, 24000.
   II. Os métodos de ordenação são: Bubble sort, Insertion sort, Selection sort, Merge sort, Quick sort, Counting sort, Radix sort, Shell sort, Bucket sort e Heapsort.
   III. Os valores armazenados nos nove vetores serão números inteiros gerados aleatoriamente.
   IV. Usar a biblioteca “matplotlib.pyplot”
   V. Plotar um gráfico comparando o tempo de execução dos algoritmos de acordo com o tamanho do vetor.

5) Implemente uma versão do heapsort que realiza ordenação decrescente. Você pode implementar em C ou C++ ou Java ou Python. Aplique essa ordenação em um vetor de 10 elementos gerados aleatoriamente.